Army Research Laboratory

A Method for Correcting Broken Hyphenations in Noisy English Text

by Jeffrey C. Micher

ARL-TN-0481 April 2012

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

Army Research Laboratory
Adelphi, MD 20783-1197

ARL-TN-0481 **April 2012**

A Method for Correcting Broken Hyphenations in Noisy English Text

Jeffrey C. Micher
Computational and Information Sciences Directorate, ARL

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*
April 2012

2. REPORT TYPE
Final

3. DATES COVERED (From - To)

4. TITLE AND SUBTITLE
A Method for Correcting Broken Hyphenations in Noisy English Text

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Jeffrey C. Micher

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army Research Laboratory
ATTN: RDRL-CII-T
2800 Powder Mill Road
Adelphi, MD 20783-1197

8. PERFORMING ORGANIZATION REPORT NUMBER
ARL-TN-0481

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The problem of rejoining broken hyphenations in processed English text is addressed. A basic algorithm is developed, which makes use of a word validation step. Results of running the algorithm over an English military training text is presented and analyzed. Precision and recall scores show that the algorithm works well for correcting broken hyphenations, but fails when certain types of noise are encountered in the data.

15. SUBJECT TERMS
hyphenation , natural language processing, OCR

16. SECURITY CLASSIFICATION OF:

a. REPORT	b. ABSTRACT	c. THIS PAGE	17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES
Unclassified	Unclassified	Unclassified	UU	18

19a. NAME OF RESPONSIBLE PERSON
Jeffrey C. Micher

19b. TELEPHONE NUMBER (*Include area code*)
(301) 394-0316

Contents

List of Tables

1. Introduction

Processing of text data plays an important role in many natural language software applications, ranging from the simplest spelling and grammar checking programs to areas such as machine translation, optical character recognition, speech recognition, and information retrieval. These applications rely on natural language data, which most often are “noisy,” i.e., they contain substantial errors or variation, and many techniques have been developed to clean up the noisiness of natural language data. One type of noisiness that occurs in processed English text is the phenomenon of broken hyphenations.

A broken hyphenation is defined as a hyphenated word, which is broken into two parts with an intervening whitespace. Examples of broken hyphenations are the following:

> specific equipment or systems
> face-to- face

Broken hyphenations occur in text that has been transferred, either automatically or manually, from one medium to another or from one text processing program to another. Once the transfer is complete, there is usually some type of processing to remove line breaks and restore sentences, but this process often does not take into account hyphenated words that had been broken apart at line breaks and thus creates broken hyphenations in the middle of sentences. For example, an optical character recognition (OCR) program may introduce broken hyphenations as it converts image data to text data. Broken hyphenations can also be introduced into text data when manually copying and pasting from one program to another simply because they are overlooked, or in the case of automatic rejoining of sentences, because the sentence rejoining algorithm neglects to take them into account.

The simple solution to this problem would be to remove a hyphen at the end of a line when rejoining the line back together or remove any hyphen followed by a space character. However, this simple solution will not always be effective because of the ambiguous nature of hyphen usage. In English, hyphens are used for two main purposes: (1) to justify a line when there is not enough space at the end of the line for a complete word, in which case, a hyphen is introduced where the line-final word is broken apart, and (2) for joining words to form compounds (*1*). Ambiguity is introduced when these two usages occur at the same time: a compound hyphenated word occurring at the end of a line will simply be split at the hyphen without adding an additional hyphen to indicate a break in a word. The following is an example:

> “… requirements for strategic-
> level planning…”

In this example, “strategic-” should be joined to “level” without removing the hyphen.

The problem is made even more complicated by “hanging” hyphens. A hanging hyphen is used in an elliptical construction of a conjunction of hyphenated terms (*1*), in which each term has the same second element, which is dropped in all but the last term. It looks exactly like a broken hyphenation in that the hyphen is followed by a space character:

> “first- and second-order planning”

A hanging hyphen should never be joined to the conjunctive item following it.

Given the various usages of hyphens listed above, the solution for rejoining broken hyphenations becomes non-trivial.

2. The Rejoining Algorithm

In order to overcome the ambiguity that hyphenation introduces into English text, a more thorough and effective solution to the problem of how to correctly rejoin broken hyphenations requires a way to validate the proposed rejoined words. The validation can be accomplished by means of a spell checking program or a large list of valid words, such as a frequency list. An algorithm that would make use of word validation, taking into account the various usages of hyphens in English, is described in pseudocode below:

```
Scan thru the text and identify all broken hyphenations.
  If the second word is ‘and,’ ‘or,’ or ‘,’, do nothing.
  Else, join the word before the hyphen with the word following the hyphen,
  removing the hyphen and the space.
    Check if this new, rejoined word is in the list of valid words.
       If so, keep it joined.
       If not, check if the two (or more) pieces can be validated
separately by looking for the pieces in the validation list.
              If true, rejoin the pieces, leaving the hyphen intact.
              If not, do not rejoin it.
Continue scanning until all potential rejoins have been processed.
```

This algorithm takes three different actions when it encounters a broken hyphen: (1) drop the hyphen and attach the following word (DA), (2) leave the hyphen and attach the following word (LA), and (3) do nothing (DN). These three categories of actions are used to evaluate the effectiveness of this algorithm.

For the current project, this algorithm was implemented in a Perl script, which is included at the end of this report in the appendix. The algorithm was run over a corpus of English text containing broken hyphenations. Below, the English corpus that was processed and the data used for the validation step are described first. Following that, empirical results of running the algorithm and an analysis of these results are presented.

3. Data Sources

The data that motivated the need for a strategy for fixing broken hyphenations is the English half of a parallel English and Arabic military training materials corpus (see the tech note on the provenance and processing of this corpus [*3*]). This corpus consists of text from training materials, such as field manuals, slide presentations, questioning lists, and Arabic language instructional materials. These data had been extracted from the original documents by hand and during this extraction process the broken hyphenations were overlooked. The processing of this corpus started with manually recombining sentences and re-segmenting the text at sentence boundaries. Then, all the words were converted to lowercase and the text was tokenized, separating punctuation such as periods, commas, and question marks from the surrounding words.

The British National Corpus (*2*) (BNC) frequency list was used to perform the validation, rather than a separate spell checking program. This was primarily because implementation of the algorithm using a frequency list was quite trivial and this frequency list was easily obtainable via the Web. Although there are spelling differences between British and American English, this did not seem to affect the outcome of correcting the broken hyphenations. The BNC contains approximately 100 million words. The frequency list derived from it contains 208,656 individual tokens.

4. Preliminary Data Analysis

A preliminary analysis of the military training corpus identified 812 broken hyphenations that potentially needed to be corrected. Of these, 607 fell into the DA category, 44 fell into LA category, and the remaining 161 fell into the DN category. A further description of the types of items in each of these categories is given below:

- DA: These were all cases of hyphenations that occurred because of line breaks.
- LA: Most of these were cases of originally hyphenated words that got broken at line breaks. Only two of these were hyphenated numerals, “fm 3- 0” and “fm 3- 93,” that got broken apart at a line break.

- DN: Forty-one of these cases were hanging hyphens. The remaining cases fell under three basic noise categories: non-standard enumeration formatting, non-standard bullets, and incorrect pause hyphens:
    - *Non-standard enumeration formatting*: 101 items consisted of a single letter or a single or double digit number followed immediately by a hyphen. Traditional formatting uses a period instead of a hyphen for enumerated lists. The following are examples of this formatting:

      b- a unit of our special forces…

      3- a standard that…
    - *Non-standard bullets*: There were a few cases of double hyphens being used as bullets, such as the following:

      -- 8 combat helmets
    - *Incorrect pause hyphens*: These were hyphens used to indicate pauses, but with the hyphen erroneously joined to the first token, as shown in the following examples:

      lesson six- radio logs

      slide 17- here we have…

      eleven men-- the squad leader and 10 squad members

In order to re-hyphenate the largest number of words while keeping the algorithm relatively simple, these cases of noise were ignored. Cleaning up this type of noise would fall under the rubric of "normalization," which is out of the scope of this study. Indeed, much of this noise was specific to this dataset so any attempt at correcting hyphenations introduced by this noise would not necessarily generalize to other data.

5. Results

The results of running the rejoining algorithm over the military training corpus are presented in table 1. The table displays the system performance on the *x*-axis and the ground truth on the *y*-axis for the three categories of actions that the algorithm should perform. The diagonals represent the items for which the system performance matched the ground truth.

Table 1. Algorithm performance.

		System		
		DA	**LA**	**DN**
Ground Truth	DA	596	2	9
	LA	2	41	1
	DN	11	101	49

The empirical results presented in table 1 can also be characterized in terms of the common metrics used in information retrieval: precision and recall. In information retrieval, precision is the number of items correctly retrieved out of all of the items retrieved. Recall is the number of items retrieved out of those that should have been retrieved. In this study, precision and recall are defined as follows: for each action (DA, LA, DN), precision indicates the number of correct actions, or "hits" out of all of the actions performed (System), and recall is the number of hits out of the number that should have been performed (ground truth [GT]). Hits are defined as those actions where the system performance matches the ground truth (System = GT). The table 1 data lends itself readily to computing these calculations. The cells in the diagonal from top left to bottom right represent the hits, for each of the three hyphenation actions. Precision for each action is calculated by dividing the number of hits by the total for that column. Recall is calculated by dividing the number of hits by the total for that row. The precision and recall scores for each of the three actions are given in table 2.

Table 2. Performance metrics.

	DA	**LA**	**DN**
Precision	0.9787	0.2847	0.8305
Recall	0.9819	0.9318	0.3043

6. Analysis

The DA precision and recall numbers, at over 97%, are quite good. The slightly lower precision score is due to a small group of misspelled words and rare usage of words, which happened not to be in the frequency list, e.g., "militaries." The LA recall is over 90%. The rejoining algorithm was able to identify 41 cases of previously hyphenated words and correctly re-hyphenate them. The LA precision value, however, was decreased due to the noise (especially the non-standard enumeration formatting) since single letters and numbers are found in the frequency list and treated as valid word parts by the algorithm. The DN precision was reasonable, having been impacted by the same issues that lowered the DA precision score. The DN recall score was extremely low and was also affected by the noisy data.

7. Conclusion

The rejoining algorithm worked well for correcting broken hyphenations. Both the DA precision and recall were high, >97%, corresponding to the cases of words hyphenated at line breaks. The LA recall was also good (≅93%), corresponding to the algorithm's ability to handle previously hyphenated words. Furthermore, the algorithm was able to identify hanging hyphens and correctly leave them alone. However, the algorithm did poorly when attempting to process noise, made up of formatting, misspellings, or numbers that had been broken apart. A noise normalization step would greatly improve the results of re-hyphenation; however, this type of processing falls outside of the scope of this study.

8. References

1. Wikipedia. “Hyphen.” http://en.wikipedia.org/wiki/Hyphen (accessed Jan 2012).
2. British National Corpus, 26 January 2009. http://www.natcorp.ox.ac.uk/ (accessed Jan 2012).
3. Micher, J. *Provenance and Processing of an English-Arabic Military Domain Corpus*; ARL technical report; U.S. Army Research Laboratory: Adelphi, MD, in preparation.

INTENTIONALLY LEFT BLANK.

Appendix. Rejoining Algorithm

The following is the Perl script for the rejoining algorithm.

```
#!/usr/bin/perl

open(FREQ, $ARGV[0]) || die "Could not open $ARGV[0]\n";

while(<FREQ>) {
   $count++;
   chop;
   $freq{$_} = $count;
}

while (<STDIN>) {
   $linenumber++;
   chop;
   $line = $_;
   @words = split(/ /,$line);
   for ($i = 0; $i < @words-1; $i++) {
      if ($words[$i] =~ /-$/ && !($words[$i+1] eq "and" ||
      $words[$i+1] eq "or" || $words[$i+1] eq ",")) {
         if ($` ne "") {
            #try to join and check freq list
            $test = $`.$words[$i+1];
            if ($freq{$test} ne "") {
               #fix by joining pieces without the hyphen
               push(@newwords, $`.$words[$i+1]);
               $i++;
            } else {
               @pieces = split(/-/,$`);
               push(@pieces, split(/-/,$words[$i+1]));
               $bool = 1; # set to true
               foreach $p (@pieces) {
                  if ($freq{$p} eq "") {#if any not in freq list,
                  whole thing fails
                     $bool = 0
                  }
               }
               if ($bool) {# check pieces, if in list
                  push(@newwords, $`."-".$words[$i+1]);
                  $i++;
               } else {#reject
                  push(@newwords, $words[$i]);
                  push(@newwords, $words[$i+1]);
                  $i++;
               }
            }
         } else { #it is a single hyphen, so leave it alone
            push(@newwords, $words[$i]);
         }
      } else {#if you don't match a hypen at end of word
         push(@newwords, $words[$i]);
      }
   }

   while(@newwords) {
      $word = shift(@newwords);
      $newline .= $word." ";
   }

   $newline =~ s/\x$//;
   print "$newline\n";
   $newline = "";
   print "$newline\n";
   $newline = "";
}
```

1 DEFENSE TECHNICAL
(PDF INFORMATION CTR
only) DTIC OCA
8725 JOHN J KINGMAN RD
STE 0944
FORT BELVOIR VA 22060-6218

1 DIRECTOR
US ARMY RESEARCH LAB
IMNE ALC HRR
2800 POWDER MILL RD
ADELPHI MD 20783-1197

1 DIRECTOR
US ARMY RESEARCH LAB
RDRL CIO LL
2800 POWDER MILL RD
ADELPHI MD 20783-1197

1 DIRECTOR
US ARMY RESEARCH LAB
RDRL CIO LT
2800 POWDER MILL RD
ADELPHI MD 20783-1197

1 DIRECTOR
US ARMY RESEARCH LAB
RDRL D
2800 POWDER MILL RD
ADELPHI MD 20783-1197

12 DIRECTOR
US ARMY RESEARCH LAB
RDRL CII B BROOME
RDRL CII T C VOSS
RDRL CII T V M HOLLAND
RDRL CII T S LAROCCA
RDRL CII T R HOBBS
RDRL CII T D BRIESCH
RDRL CII T L HERNANDEZ
RDRL CII T J MICHER (5 COPIES)
2800 POWDER MILL RD
ADELPHI MD 20783-1197

INTENTIONALLY LEFT BLANK.